Amendments to the Specification

Please replace paragraph [0014] with the following amended paragraph:

[0014] The present invention solves the scalability problem of global address space (GAS) languages by utilizing a layered approach wherein a compiler parses the global address space code and transforms the global address space language constructs into calls to a scalable runtime system. The runtime system consists of a set of data structures and functions that operate on these data structures. The runtime system implements the global address space language semantics on top of the hardware and operating system primitives. The interface exposed by the runtime system is applicable to both ~~share~~ shared and distributed memory machines. While the embodiments of the invention discussed below are directed toward the UPC global address space language, it can be readily applied to all global address space and similar languages.

Please replace paragraph [0020] with the following amended paragraph:

[0020] Referring now to FIG. 2, a method of creating a scalable runtime system for a global address space language is shown. The method begins in step 202 with the providing of a partition associated with each thread of a program having handles to shared data. A partition is also provided for statically declared non-scalar variables in step 204. A directory of shared variables having an array of pointers to the partitions is established in step 206. To create shared variables, program threads allocate entries in the directory of shared variables in step 208. Since the owning thread allocates the entries, there is no need to synchronize the allocation on a shared memory configuration. In step 210, on a distributed memory machine, the directory of shared variables partitions are updated in a consistent manner. In step 212, the shared data is accessed through the directory of shared variables with a partition index and a variable index. When a thread ~~threads~~ calls on a variable, the partition for which the variable has

affinity is dereferenced as set forth in step 214. Once the partition is dereferenced, a control data structure is used to access the shared memory portion as set forth in step 216.